Asynchronous Doubly Stochastic Group Regularized Learning

Bin Gu **Zhouyuan Huo** **Heng Huang**
Department of Electrical and Computer Engineering, University of Pittsburgh, USA

Abstract

Group regularized learning problems (such as group Lasso) are important in machine learning. The asynchronous parallel stochastic optimization algorithms have received huge attentions recently as handling large scale problems. However, existing asynchronous stochastic algorithms for solving the group regularized learning problems are not scalable enough simultaneously in sample size and feature dimensionality. To address this challenging problem, in this paper, we propose a novel asynchronous doubly stochastic proximal gradient algorithm with variance reduction (AsyDSPG+). To the best of our knowledge, AsyDSPG+ is the first asynchronous doubly stochastic proximal gradient algorithm, which can scale well with the large sample size and high feature dimensionality simultaneously. More importantly, we provide a comprehensive convergence guarantee to AsyDSPG+. The experimental results on various large-scale real-world datasets not only confirm the fast convergence of our new method, but also show that AsyDSPG+ scales better than the existing algorithms with the sample size and dimension simultaneously.

1 Introduction

Group regularized learning problems are important in machine learning. Take group Lasso [Roth and Fischer, 2008] as an example, when the features are partitioned into groups, the group lasso penalty (also called ℓ_1/ℓ_2 penalty) tends to select features in a grouped manner. Formally, given a partition $\{\mathcal{G}_1, \cdots, \mathcal{G}_k\}$ of n features (i.e., coordinates) of x, we can write the group regularized penalty function $g(x)$ as $g(x) = \sum_{j=1}^{k} g_{\mathcal{G}_j}(x_{\mathcal{G}_j})$. In this paper, we focus on the composite group regularized optimization problem as follows:

$$\min_{x\in\mathbb{R}^n} F(x) = f(x) + g(x) \tag{1}$$

where $f(x) = \frac{1}{l}\sum_{i=1}^{l} f_i(x)$, $f_i : \mathbb{R}^n \mapsto \mathbb{R}$ is a smooth, possibly non-convex function. $g : \mathbb{R}^n \mapsto \mathbb{R} \cup \{\infty\}$ is a block separable, closed, convex, and extended real-valued function as defined above. The formulation (1) covers an extensive number of group regularized learning problems, such as group Lasso [Roth and Fischer, 2008], block dictionary learning [Chi et al., 2013]) and sparse multi-class classification [Blondel et al., 2013].

In the current big data era, asynchronous parallel algorithms for stochastic optimization have received huge successes in theory and practice recently. Specifically, for the problem (1) with $g(x) = 0$, Recht et al. [Recht et al., 2011] proposed the first asynchronous parallel stochastic gradient descent (SGD) algorithm known as Hogwild!. Zhao and Li [2016], Huo and Huang [2017] proposed asynchronous parallel SGD algorithms with the SVRG variance reduction technique. Mania et al. [2015] proposed a perturbed iterate framework to analyze the asynchronous parallel algorithms of SGD, stochastic coordinate descent (SCD). For the problem (1) with $g(x) = \frac{1}{2}\|x\|^2$, Hsieh et al. [2015] proposed an asynchronous parallel stochastic dual coordinate descent algorithm. For the problem (1) with a separable function $g(x)$ in coordinate, Liu and Wright [2015] proposed an asynchronous stochastic coordinate descent (SCD) algorithm. These works are summarized in Table 1. From Table 1, it is easy to find that these asynchronous stochastic algorithms cannot solve the composite group regularized problem (1).

To the best of our knowledge, there are several (asynchronous parallel) stochastic algorithms can solve the problem (1). These works are summarized in Table 1. Specifically, Hong et al. [2013] proposed a batch randomized block coordinate descent method which runs with full gradient on the randomized block coordinates. Zhao et al. [2014] proposed a accelerated double stochastic proximal gradient algorithm (DSPG) with the SVRG variance reduction technique, which is stochastic on samples and coordinates simultaneously

Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS) 2018, Lanzarote, Spain. PMLR: Volume 84. Copyright 2018 by the author(s).

Table 1: Representative (asynchronous) stochastic algorithms.

Reference	$g(x)$	**Stochastic**	**Asynchronous**	**Accelerated**	**Solve (1)**
Recht et al. [2011]	0	Samples	Yes	No	No
Zhao and Li [2016]	0	Samples	Yes	Yes	No
Mania et al. [2015]	0	Samples/Coordinates	Yes	No	No
Hsieh et al. [2015]	$\frac{1}{2}\|x\|^2$	Coordinates	Yes	No	No
Liu and Wright [2015]	Separable	Coordinates	Yes	No	No
Hong et al. [2013]	Block Separable	Block coordinates	No	No	Yes
Zhao et al. [2014]	Block Separable	Samples+Coordinates	No	Yes	Yes
Li et al. [2013, 2016]	$g(x)$	Samples	Yes	No	Yes
Meng et al. [2016]	$g(x)$	Samples	Yes	Yes	Yes
Our	Block Separable	Samples+Coordinates	Yes	Yes	Yes

for each iteration. Li et al. [2013] and Li et al. [2016] proposed asynchronous parallel stochastic proximal optimization algorithms. Meng et al. [2016] proposed an asynchronous parallel stochastic proximal optimization algorithm with the SVRG variance reduction technique.

However, existing (asynchronous parallel) stochastic algorithms as mentioned above are not scalable enough simultaneously in sample size and feature dimensionality, for solving the group regularized learning problem (1). In the big data era, a lot of machine learning applications (such as the movie recommendation [Diao et al., 2014], the advertisement recommendation [Joachims and Swaminathan, 2016], the computer aided diagnostic [Bi et al., 2006] and so on) need to solve the large-scale optimization problems, where both of the sample size and dimension could be huge at the same time. As shown in Table 1, most of existing (asynchronous parallel) stochastic algorithms are stochastic either on samples or block coordinates. Thus, these (asynchronous parallel) stochastic algorithms [Hong et al., 2013, Li et al., 2013, 2016, Meng et al., 2016] cannot scale well in sample size and dimensionality simultaneously. Note that, although Zhao et al. [2014] proposed a doubly stochastic proximal gradient algorithm to solve the problem (1), they did not utilize the asynchronous parallel computation technique which is extremely important for the large-scale optimization. Due to the inconsistent reading and writting induced by asynchronous parallel computation, it is not trivial to give the asynchronous doubly stochastic proximal gradient algorithm with the corresponding theoretical guarantee.

To address this challenging problem, we propose an asynchronous doubly stochastic proximal gradient algorithm with the SVRG variance reduction technique (AsyDSPG+). To the best of our knowlege, AsyDSPG+ is the first asynchronous doubly stochastic proximal gradient algorithm, which can scale well with the large sample size and high feature dimensionality simultaneously. More importantly, we prove that AsyDSPG+ achieves a linear convergence rate when the function f has the optimal strong convexity property, and a sublinear rate when f is with the general convexity or with the non-convexity. The experimental results on various large-scale real-world datasets not only confirm the fast convergence of our new method, but also show that AsyDSPG+ scales better than the existing algorithms with the sample size and dimension simultaneously. We also used AsyDSPG+ to implement large scale sparse kernel learning [Bin Gu, 2018b].

Notations. To make the paper easier to follow, we give the following notations.

- Δ_j denote the zero vector in $\mathbb{R}^n$ except that the block coordinates indexed by the set $\mathcal{G}_j$.
- $\|\cdot\|$ denotes the Euclidean norm.
- $\nabla f(x)$ is the gradient of function $f(\cdot)$ at the point x.
- $\mathcal{P}_{j,g_j}(x')$ is the blockwise proximal operator as $\mathcal{P}_{j,g_j}(x') = \arg\min_w \frac{1}{2}\|x - x'\|^2 + h\left((x)_{\mathcal{G}_j}\right)$.
- $\mathcal{P}_S(x)$ is the Euclidean-norm projection of a vector x onto a given set S as $\mathcal{P}_S(x) = \arg\min_{y \in S} \|y - x\|^2$.

2 AsyDSPG+ Algorithm

In this section, we first give a brief discussion on the existing algorithms for solving the group regularized problem (1), and then propose our AsyDSPG+ algorithm.

2.1 Brief Discussion on Existing Algorithms

As shown in Table 1, existing algorithms for solving the group regularized problem (1) did not utilize the techniques of doubly stochastic gradient, variance reduction and asynchronous parallel computation, which are the important techniques for large-scale optimization.

Doubly stochastic gradient: Most of existing algorithms are single stochastic on samples or coordinates which cannot scale well in sample size and dimensionality simultaneously. The only algorithm with doubly stochastic gradient is the work of [Zhao et al., 2014].

Accelerated technique of variance reduction: For stochastic gradient algorithms, the large variance on gradient is the fundamental reason of slow convergence rate. Thus, various accelerated techniques of variance reduction have been used to accelerate the stochastic gradient algorithms. For solving (1), the works of [Zhao et al., 2014] and [Meng et al., 2016] used the SVRG variance reduction technique.

Asynchronous parallel computation: The parallel computation as the basic big data technique can be roughly divided into synchronous and asynchronous models according to whether the reading or writing lock is used. The asynchronous computation is much more efficient than the synchronous computation, because it keeps all computational resources busy all the time. For solving (1), the works of [Li et al., 2013, 2016, Meng et al., 2016] used the asynchronous parallel technique.

In this paper, we want to design a new algorithm for solving the group regularized problem (1) by utilizing the above techniques comprehensively.

2.2 Our AsyDSPG+ Algorithm

AsyDSPG+ is designed for the parallel environment with shared memory, such as multi-core processors and GPU-accelerators, but it can also work in the parallel environment with distributed memory.

In the parallel environment with shared memory, all cores in CPU or GPU can read and write the vector x in the shared memory simultaneously without any lock. Besides randomly choosing a sample set and a block of coordinates, AsyDSPG+ is also accelerated by the variance reduction. Thus, AsyDSPG+ has two-layer loops. The outer layer is to parallelly compute the full gradient $\nabla f(x^s) = \frac{1}{l}\sum_{i=1}^{l} \nabla f_i(x^s)$, where the superscript s denotes the s-th outer loop. The inner layer is to parallelly and repeatedly update the vector x in the shared memory. Specifically, all cores repeat the following steps independently and concurrently without any lock:

1. **Read:** Read the vector x from the shared memory to the local memory without reading lock. We use $\widehat{x}_t^{s+1}$ to denote its value, where the subscript t denotes the t-th inner loop.

2. **Compute:** Randomly choose a mini-batch $\mathcal{B}$ and a block coordinate j from $\{1, ..., k\}$, and locally compute $\widehat{v}_{\mathcal{G}_j}^{s+1} = \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} \left(\nabla_{\mathcal{G}_j} f_i(\widehat{x}_t^{s+1}) - \nabla_{\mathcal{G}_j} f_i(\widetilde{x}^s)\right) + \nabla_{\mathcal{G}_j} f(\widetilde{x}^s)$.

3. **Update:** Update the block j of the vector x in the shared memory as $x_{t+1}^{s+1} \leftarrow \mathcal{P}_{\mathcal{G}_j, \frac{\gamma}{L_{\max}} g_j}((x_t^{s+1})_{\mathcal{G}_j} - \frac{\gamma}{L_{\max}} \widehat{v}_{\mathcal{G}_j}^{s+1})$ without writing lock.

The detailed description of AsyDSPG+ is summarized in Algorithm 1. Note that $\widehat{v}_{\mathcal{G}_j}^{s+1}$ computed locally is the approximation of $\nabla_{\mathcal{G}_j} f(\widehat{x}_t^{s+1})$, and the expectation of $\widehat{v}_t^{s+1}$ on $\mathcal{B}$ is equal to $\nabla f(\widehat{x}_t^{s+1})$ as follows:

$$\begin{aligned} & \mathbb{E}\widehat{v}_t^{s+1} \qquad (2)\\ = \ & \nabla f(\widehat{x}_t^{s+1}) - \nabla f(\widetilde{x}^s) + \nabla f(\widetilde{x}^s) = \nabla f(\widehat{x}_t^{s+1}). \end{aligned}$$

Thus, $\widehat{v}_t^{s+1}$ is called an unbiased stochastic gradient of $f(x)$ at $\widehat{x}_t^{s+1}$.

Algorithm 1 Asynchronous Doubly Stochastic Proximal Gradient Algorithm with Variance Reduction (AsyDSPG+)

```
Input: The number of outer loop iterations $S$, the number of inner loop iterations $m$, and learning rate $\gamma$.
Output: $x^S$.
 1: Initialize $x^0 \in \mathbb{R}^d$, $p$ threads.
 2: for $s = 0, 1, 2, \cdots, S-1$ do
 3:   $\widetilde{x}^s \leftarrow x^s$
 4:   All threads parallelly compute the full gradient $\nabla f(\widetilde{x}^s) = \frac{1}{l}\sum_i^l \nabla f_i(\widetilde{x}^s)$
 5:   For each thread, do:
 6:   for $t = 0, 1, 2, \cdots, m-1$ do
 7:     Randomly sample a mini-batch $\mathcal{B}$ from $\{1, ..., l\}$ with equal probability.
 8:     Randomly choose a block $j(t)$ from $\{1, ..., k\}$ with equal probability.
 9:     Compute $\widehat{v}_{\mathcal{G}_{j(t)}}^{s+1} \leftarrow \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} \nabla_{\mathcal{G}_j} f_i(\widehat{x}_t^{s+1}) - \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} \nabla_{\mathcal{G}_j} f_i(\widetilde{x}^s) + \nabla_{\mathcal{G}_{j(t)}} f(\widetilde{x}^s)$.
10:     $x_{t+1}^{s+1} \leftarrow \mathcal{P}_{\mathcal{G}_{j(t)}, \frac{\gamma}{L_{\max}} g_{j(t)}}((x_t^{s+1})_{\mathcal{G}_{j(t)}} - \frac{\gamma}{L_{\max}} \widehat{v}_{t,\mathcal{G}_{j(t)}}^{s+1})$.
11:   end for
12:   $x^{s+1} \leftarrow x_m^{s+1}$
13: end for
```

3 Preliminaries

In this section, we introduce the condition of optimal strong convexity and three different Lipschitz constants, and give the corresponding assumptions, which are all critical to the analysis of AsyDSPG+.

Optimal Strong Convexity: Let F^* denote the optimal value of (1), and let S denote the solution set of F such that $F(x) = F^*$, $\forall x \in S$. Firstly, we assume that S is nonempty (i.e., Assumption 1), which is reasonable to the problem (1).

Assumption 1. *The solution set S of the problem (1) is nonempty.*

Then, we assume that the convex function f is with the optimal strong convexity (*i.e.*, Assumption 2).

Assumption 2 (Optimal strong convexity)**.** *The convex function f has the condition of optimal strong convexity with parameter $l > 0$ with respect to the optimal set S, which means that, $\exists l$ such that, $\forall x$, we have:*

$$F(x) - F(\mathcal{P}_S(x)) \geq \frac{l}{2}\|x - \mathcal{P}_S(x)\|^2 . \tag{3}$$

As mentioned in [Liu and Wright, 2015], the condition of optimal strong convexity is significantly weaker than the normal strong convexity condition. Several examples of optimally strongly convex functions that are not strongly convex are provided in [Liu and Wright, 2015].

Lipschitz Smoothness: We define the normal Lipschitz constant (L_{nor}), block restricted Lipschitz constant (L_{res}) and block coordinate Lipschitz constant ($L_{\max}$) as follows.

Definition 1 (Normal Lipschitz constant)**.** *L_{nor} is the normal Lipschitz constant for ∇f_i ($\forall i \in \{1, \cdots, l\}$) in (1), such that, $\forall x$ and $\forall y$, we have:*

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L_{nor}\|x - y\| . \tag{4}$$

Definition 2 (Block restricted Lipschitz constant)**.** *L_{res} is the block restricted Lipschitz constant for ∇f_i ($\forall i \in \{1, \cdots, l\}$) in the problem (1), such that, $\forall x$, and $\forall j \in \{1, \cdots, k\}$, we have:*

$$\|\nabla f_i(x + \Delta_j) - \nabla f_i(x)\| \leq L_{res} \left\|(\Delta_j)_{\mathcal{G}_j}\right\| . \tag{5}$$

Definition 3 (Block coordinate Lipschitz constant)**.** *$L_{\max}$ is the block coordinate Lipschitz constant for ∇f_i ($\forall i \in \{1, \cdots, l\}$) in the problem (1), such that, $\forall x$, and $\forall j \in \{1, \cdots, k\}$, we have:*

$$\max_{j=1,\cdots,k} \|\nabla f_i(x + \Delta_j) - \nabla f_i(x)\| \leq L_{\max} \left\|(\Delta_j)_{\mathcal{G}_j}\right\| . \tag{6}$$

The inequality (6) can be re-written as the following formulation:

$$\begin{aligned} & f_i(x + \Delta_j) & (7) \\ \leq \quad & f_i(x) + \langle \nabla_{\mathcal{G}_j} f_i(x), (\Delta_j)_{\mathcal{G}_j} \rangle + \frac{L_{\max}}{2} \left\|(\Delta_j)_{\mathcal{G}_j}\right\|^2 . \end{aligned}$$

Based on L_{nor}, L_{res}, and L_{max} as defined above, we assume that the function f_i ($\forall i \in \{1, \cdots, l\}$ is Lipschitz smooth with L_{nor}, L_{res}, and L_{max} (*i.e.*, Assumption 3). In addition, we define $\Lambda_{res} = \frac{L_{res}}{L_{\max}}$, $\Lambda_{nor} = \frac{L_{nor}}{L_{\max}}$.

Assumption 3 (Lipschitz smoothness)**.** *The function f_i ($\forall i \in \{1, \cdots, l\}$ is Lipschitz smooth with the normal Lipschitz constant L_{nor}, block restricted Lipschitz constant L_{res} and block coordinate Lipschitz constant $L_{\max}$.*

4 Convergence Analysis

In this section, we prove the convergence rates of AsyDSPG+. To the best of our knowledge, this is the first work to prove the convergence rates for the asynchronous doubly stochastic proximal gradient algorithm.

4.1 Difficulties for the Analysis

We first pointed out there exist two difficulties for proving the convergence rates of AsyDSPG+. Specifically, the difficulties are summarized as follows.

1. Normally, the variance of $\|x_{t-1}^s - x_t^s\|^2$ for the doubly stochastic gradient algorthms is larger than the one for the single stochastic gradient algorithm. There would have higher variance of $\|x_{t-1}^s - x_t^s\|^2$ for our AsyDSPG+, because the gradient of our AsyDSPG+ is computed on the randomly selected samples and the randomly selected block of coordinates.
2. AsyDSPG+ runs asynchronously without any lock, the *inconsistent reading and writting* could arise to the vector x in the shared memory, which would make the values of x in the shared memory and local memory inconsistent.

Due to the these complications, it is not trivial to provide the the convergence analysis for AsyDSPG+. In the following, we will address these two difficulties.

4.1.1 First Difficulty

To address the first challenge, we give a bound to $\|\mathbb{E}_{j(t)}(x_{t+1}^s - x_t^s)\|^2$. Specifically, we first define a new vector $\overline{x}_{t+1}^s$ without the stochasticness on coordinates. Specifically, $\overline{x}_{t+1}^s$ is defined as follows.

$$\overline{x}_{t+1}^s \overset{\text{def}}{=} \mathcal{P}_{\frac{\gamma}{L_{\max}} g}\left(x_t^s - \frac{\gamma}{L_{\max}} \widehat{v}_t^s\right) . \tag{8}$$

Based on Eq. (8), it is easy to verify that $(\overline{x}_{t+1}^s)_{\mathcal{G}_{j(t)}} = (x_{t+1}^{s+1})_{\mathcal{G}_{j(t)}}$. Thus, we have:

$$\mathbb{E}_{j(t)}(x_{t+1}^s - x_t^s) = \frac{1}{k}\left(\overline{x}_{t+1}^s - x_t^s\right) \tag{9}$$

It means that $\overline{x}_{t+1}^s - x_t^s$ captures the expectation of $x_{t+1}^s - x_t^s$ on the coordinates. Based on $\overline{x}_{t+1}^s$, we give an upper bound of $\mathbb{E}\|x_{t-1}^s - \overline{x}_t^s\|^2$ as $\mathbb{E}\|x_{t-1}^s - \overline{x}_t^s\|^2 \leq \rho\mathbb{E}\|x_t^s - \overline{x}_{t+1}^s\|^2$ (Lemma 1), where $\rho > 1$ is a user defined parameter.

Lemma 1. *Let ρ be a constant that satisfies $\rho > 1$, and define the quantities $\theta_1 = \frac{\rho^{\frac{1}{2}} - \rho^{\frac{\tau+1}{2}}}{1-\rho^{\frac{1}{2}}}$ and $\theta_2 = \frac{\rho^{\frac{1}{2}} - \rho^{\frac{m}{2}}}{1-\rho^{\frac{1}{2}}}$. Suppose the nonnegative* *step length parameter $\gamma > 0$ satisfies: $\gamma \leq \min\left\{ \frac{k^{1/2}(1-\rho^{-1})-4}{4(\Lambda_{res}(1+\theta_1)+\Lambda_{nor}(1+\theta_2))}, \frac{k^{1/2}}{\frac{1}{2}k^{1/2}+2\Lambda_{nor}\theta_2+\Lambda_{res}\theta_1} \right\}$, under Assumptions 1, 3 and 4, we have*

$$\mathbb{E}\|x_{t-1}^s - \overline{x}_t^s\|^2 \leq \rho \mathbb{E}\|x_t^s - \overline{x}_{t+1}^s\|^2 \tag{10}$$

The detailed proof of Lemma 1 is omitted here due to the space limit.

4.1.2 Second Difficulty

Because AsyDSPG+ does not use the reading lock, the vector $\widehat{x}_t^{s+1}$ read into the local memory may be inconsistent to the vector x_t^{s+1} in the shared memory, which means that some components of $\widehat{x}_t^{s+1}$ are same with the ones in x_t^{s+1}, but others are different to the ones in x_t^{s+1}. We call this as inconsistent reading. To address the challenge of inconsistent reading, we assume an upper bound to the delay of updating. Specifically, we define a set $K(t)$ of inner iterations, such that:

$$x_t^{s+1} = \widehat{x}_t^{s+1} + \sum_{t' \in K(t)} B_{t'}^{s+1} \Delta_{t'}^{s+1}, \tag{11}$$

where $t' \leq t-1$, $(\Delta_{t'}^{s+1})_{\mathcal{G}_{j(t')}} = \mathcal{P}_{\mathcal{G}_{j(t')}, \frac{\gamma}{L_{\max}} g_{j(t')}}((x_{t'}^{s+1})_{\mathcal{G}_{j(t')}} - \frac{\gamma}{L_{\max}} \widehat{v}_{t', \mathcal{G}_{j(t')}}^{s+1}) - (x_{t'}^{s+1})_{\mathcal{G}_{j(t')}}$, $(\Delta_{t'}^{s+1})_{\setminus \mathcal{G}_{j(t')}} = \mathbf{0}$, and $B_{t'}^{s+1}$ is a diagonal matrix with diagonal entries either 1 or 0. It is reasonable to assume that there exists an upper bound τ such that $\tau \geq t - \min\{t' | t' \in K(t)\}$ (*i.e.*, Assumption 4).

Assumption 4 (Bound of delay)**.** *There exists an upper bound τ such that $\tau \geq t - \min\{t' | t' \in K(t)\}$ for all inner iterations t in AsyDSPG+.*

In addition, AsyDSPG+ does not use any writing lock. Thus, in the line 10 of Algorithm 1, x_t^s (left side of '$\leftarrow$') updated in the shared memory may be inconsistent with the ideal one (right side of '$\leftarrow$') computed by the proximal operator. We call this as inconsistent writting. To address the challenge of inconsistent writting, we use x_t^s to denote the ideal one computed by the proximal operator in the analysis. Same as mentioned in [Mania et al., 2015], there might not be an actual time the ideal ones exist in the shared memory, except the first and last iterates for each outer loop. It is noted that, x_0^s and x_m^s are exactly what is stored in shared memory. Thus, we only consider the ideal x_t^s in the analysis.

4.2 Convergence Rate Analysis

After addressing the above challenges, we provide a comprehensive convergence guarantee to AsyDSPG+, based on the basic assumptions (*i.e.*, Assumptions 1, 2, 3 and 4). Specifically, we have the following conclusions.

1. If the function $f(x)$ is with the optimal strong convexity, AsyDSPG+ achieves a linear convergence rate (Theorem 1).
2. If the function $f(x)$ is general convex, AsyDSPG+ achieves a sublinear convergence rate (Theorem 1).
3. If the function $f(x)$ is non-convex, AsyDSPG+ converges to a stationary point with a sublinear convergence rate (Theorem 2).

4.2.1 Convex Setting

Before proving the convergence rates of AsyDSPG+, we first prove the monotonicity of the expectation of the objectives $\mathbb{E}F(x_{t+1}^s) \leq \mathbb{E}F(x_t^s)$ (Lemma 2).

Lemma 2. *Let ρ be a constant that satisfies $\rho > 1$, and define the quantities $\theta_1 = \frac{\rho^{\frac{1}{2}} - \rho^{\frac{\tau+1}{2}}}{1-\rho^{\frac{1}{2}}}$ and $\theta_2 = \frac{\rho^{\frac{1}{2}} - \rho^{\frac{m}{2}}}{1-\rho^{\frac{1}{2}}}$. Suppose the nonnegative step length parameter $\gamma > 0$ satisfies: $\gamma \leq \min\left\{ \frac{k^{1/2}(1-\rho^{-1})-4}{4(\Lambda_{res}(1+\theta_1)+\Lambda_{nor}(1+\theta_2))}, \frac{k^{1/2}}{\frac{1}{2}k^{1/2}+2\Lambda_{nor}\theta_2+\Lambda_{res}\theta_1} \right\}$. Under Assumptions 1, 3 and 4, the expectation of the objective function $\mathbb{E}F(x_t^s)$ is monotonically decreasing,* i.e., *$\mathbb{E}F(x_{t+1}^s) \leq \mathbb{E}F(x_t^s)$.*

The detailed proof of Lemma 2 is omitted here due to the space limit. Based on Lemma 2, we prove that the convergence rates of AsyDSPG+ in the convex and strong convex settings for the function $f(x)$.

Theorem 1. *Let ρ be a constant that satisfies $\rho > 1$, and define the quantities $\theta_1 = \frac{\rho^{\frac{1}{2}} - \rho^{\frac{\tau+1}{2}}}{1-\rho^{\frac{1}{2}}}$, $\theta_2 = \frac{\rho^{\frac{1}{2}} - \rho^{\frac{m}{2}}}{1-\rho^{\frac{1}{2}}}$ and $\theta' = \frac{\rho^{\tau+1}-\rho}{\rho-1}$. Suppose the nonnegative step length parameter $\gamma > 0$ satisfies:*

$$1 - \Lambda_{nor}\gamma - \frac{\gamma\tau\theta'}{n} - \frac{2(\Lambda_{res}\theta_1 + \Lambda_{nor}\theta_2)\gamma}{n^{1/2}} \geq 0 \tag{12}$$

If the optimal strong convexity holds for f with $l > 0$ (i.e., Assumption 2), we have:

$$\mathbb{E}F(x^s) - F^* \leq \frac{L_{\max}}{2\gamma} \left(\frac{1}{1 + \frac{m\gamma l}{k(l\gamma + L_{\max})}} \right)^s \cdot \left(\|x^0 - \mathcal{P}_S(x^0)\|^2 + \frac{2\gamma}{L_{\max}} \left(F(x^0) - F^* \right) \right) \tag{13}$$

If f is a general smooth convex function, we have

$$\begin{aligned} &\mathbb{E}F(x^s) - F^* \\ \leq \quad & \frac{kL_{\max}\|x^0 - \mathcal{P}_S(x^0)\|^2 + 2\gamma k \left(F(x^0) - F^* \right)}{2\gamma k + 2m\gamma s} \end{aligned} \tag{14}$$

Remark 1. *Theorem 1 shows that, if the objective function $P(w)$ is with the optimal strong convexity,* *AsyDSPG+ achieves a linear convergence rate (see (13)). If the loss function $f_i(w)$ is general smooth convex, AsyDSPG+ achieves a sublinear convergence rate (see (14)).*

Remark 2. *The thread number is not explicitly considered in Theorems 1. As discussed in [Liu and Wright, 2015], the parameter τ is closely related to the number of threads that can be involved in the computation, without degrading the convergence performance of the algorithm. In other words, if the number of threads is small enough such that (12) holds, the convergence expressions (13), (14) do not depend on the number of threads, implying that linear speedup can be expected.*

4.2.2 Non-convex Setting

If the function $F(w)$ is non-convex, the global optimum point cannot be guaranteed. Thus, the closeness to the optimal solution (i.e., $F(w) - F^*$ and $\|x - \mathcal{P}_S(w)\|$) cannot be used for the convergence analysis. To analyze the convergence rate of AsyDSPG+ in the non-convex setting, we define the expectation of a subgradient $\xi \in \partial P(w_t^s)$ as $\mathbb{E}\widetilde{\nabla} F(w_t^s)$. Specifically, $\mathbb{E}\widetilde{\nabla} F(w_t^s)$ can be written as following.

$$\mathbb{E}\widetilde{\nabla} F(x_t^s) \overset{\text{def}}{=} \frac{L_{\max}}{\gamma}\left(x_t^s - \overline{x}_{t+1}^s\right) \tag{15}$$

It is easy to verify that $\mathbb{E}\widetilde{\nabla} F(x_t^s)$ is equal to $\mathbf{0}$ when AsyDSPG+ approaches to a stationary point.

Based on $\mathbb{E}\widetilde{\nabla} F(w_t^s)$, we give the convergence rate of AsyDSPG+ at the non-convex setting in Theorem 2.

Theorem 2. *Let ρ be a constant that satisfies $\rho > 1$, and define the quantities $\theta_1 = \frac{\rho^{\frac{1}{2}} - \rho^{\frac{\tau+1}{2}}}{1-\rho^{\frac{1}{2}}}$ and $\theta_2 = \frac{\rho^{\frac{1}{2}} - \rho^{\frac{m}{2}}}{1-\rho^{\frac{1}{2}}}$. Suppose the nonnegative step length parameter $\gamma > 0$ satisfies: $\gamma \leq \min\left\{\frac{k^{1/2}(1-\rho^{-1})-4}{4(\Lambda_{res}(1+\theta_1)+\Lambda_{nor}(1+\theta_2))}, \frac{k^{1/2}}{\frac{1}{2}k^{1/2}+2\Lambda_{nor}\theta_2+\Lambda_{res}\theta_1}\right\}$. Let T denote the number of total iterations of AsyDSPG+. If f is a smooth non-convex function, we have:*

$$\frac{1}{T}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\mathbb{E}\left\|\widetilde{\nabla} F(x_t^s)\right\|^2 \tag{16}$$
$$\leq \left(\frac{\gamma}{k}\left(\frac{1}{\gamma} - \frac{1}{2} - \frac{2L_{nor}\theta_2 + L_{res}\theta_1}{k^{1/2}L_{\max}}\right)\right)^{-1}\frac{F(x^0) - F^*}{T}$$

Remark 3. *Theorem 2 shows that, if the function $f(x)$ is non-convex, AsyDSPG+ converges to a stationary point with a sublinear convergence rate.*

5 Experimental Results

In this section, we first describe the experimental setup, and then provide our experimental results and discussions.

5.1 Experimental Setup

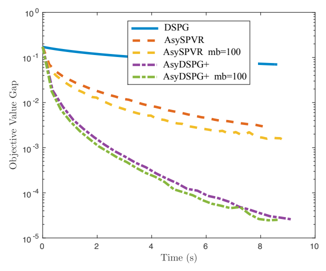
We describe our experimental setup from the following four aspects, *i.e.*, the compared methods, the solved problems, the implementation, and the datasets.

Compared methods: To verify the scalability of our AsyDSPG+, we compare the convergence of objective function of our AsyDSPG+ with the state-of-the-art (asynchronous) stochastic algorithms which can solve (1). Specifically, as mentioned in Table 1, we compare with the double stochastic block proximal descent method (DSPG) [Zhao et al., 2014] and asynchronous stochastic proximal optimization algorithm with the SVRG variance reduction technique (AsySPVR) Meng et al. [2016]. Note that DSPG and AsySPVR are the accelerated versions of the batch randomized block coordinate descent method [Hong et al., 2013] and the asynchronous stochastic proximal optimization algorithm [Li et al., 2013, 2016] respectively, we only compare with DSPG and AsySPVR. Specifically, the compared methods are

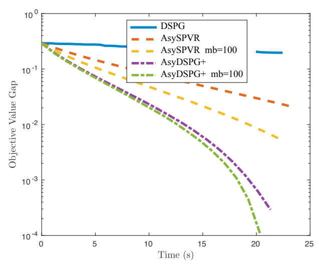
1. DSPG: DSPG [Zhao et al., 2014] is the non-parallel version of our AsyDSPG+.
2. AsySPVR: AsySPVR [Meng et al., 2016] with the mini-batch size is 1.
3. AsySPVR $mb = 100$: AsySPVR [Meng et al., 2016] with the mini-batch size 100.
4. AsyDSPG+: Our AsyDSPG+ with the mini-batch size 1.
5. AsyDSPG+ $mb = 100$: Our AsyDSPG+ with the mini-batch size 100.

To show a near-linear speedup obtained by asynchronous parallel computation, we also test the speedup of our AsyDSPG+.

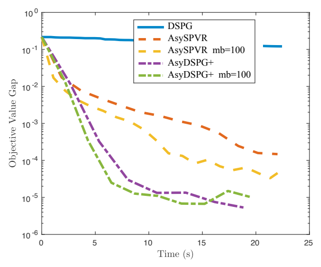
Problems: In the experiments, we consider both binary classification and regression problems. Let $S = \{(a_i, b_i)\}_{i=1}^l$ be a training set, where $a_i \in \mathbb{R}^n$, $b_i \in \{+1, -1\}$ is for binary classification, $b_i \in \mathbb{R}$ is for regression. For the function $f_i(x)$ in the problem (1), we consider the logistic loss, the sigmoid loss, and the least square loss as presented in Table 2, where the logistic loss and sigmoid loss are for binary classification, the square loss is for regression. Note that the sigmoid loss is non-convex. In the experiments, we use



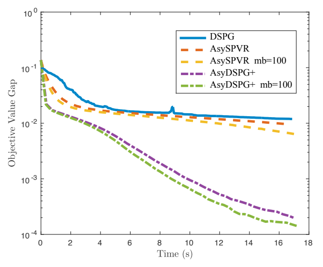
(a) Covtype_B



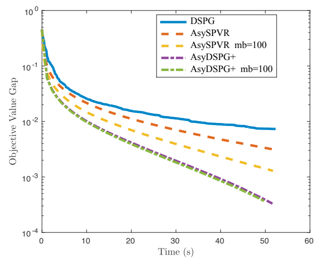
(b) RCV1



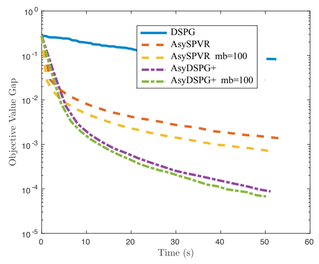
(c) SUSY



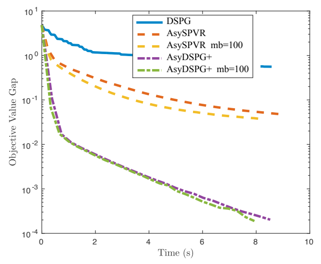
(d) Covtype_B



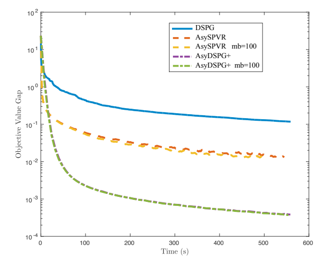
(e) RCV1



(f) SUSY



(g) Covtype_M



(h) MNIST

Figure 1: Convergence of objective value vs. running time for different algorithms. (a)-(c) Logistic loss. (d)-(f) Sigmoid loss. (g)-(h) Square loss.

Table 2: The learning problems used in our experiments (BC=binary classification, R=regression).

Type of function	Name of loss	Type of task	The loss function	Group regularization term
Convex	Logistic loss	BC	$\log(1+e^{-b_i x^T a_i})$	$\lambda \sum_{j=1}^{k} \|x_{\mathcal{G}_j}\|_2$
	Square loss	R	$(b_i - x^T a_i)^2$	
Non-convex	Sigmoid loss	BC	$\frac{1}{1+e^{b_i x^T a_i}}$	

the group regularization term $\lambda \sum_{j=1}^{k} \|x_{\mathcal{G}_j}\|_2$ as the block separable function $g(x)$ in the problem (1).

Implementation: Our experiments are performed on a 32-core two-socket Intel Xeon E5-2699 machine where each socket has 16 cores. We also call each core as worker in the experiment. We implement our AsyDSPG+ using C++, where the shared memory parallel computation is handled via OpenMP. We also implement batch randomized block coordinate descent method (BRBCD) and asynchronous stochastic block coordinate descent method (AsySPVR) in C++. In each experiment, the learning rate γ for all compared methods is selected from $\{10^2, 10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$. In the experiments, all results are the average of 5 trials.

Datasets: Table 3 summarizes the five large-scale real-world datasets used in our experiments[1]. They are the Covtype_B, RCV1, SUSY, Covtype_M and MNIST datasets, where Covtype_B and Covtype_M are from a same source. Covtype_M and MNIST are originally for multi-class classification. Note that, to obtain high dimensional data with group constraints, we duplicate the features of the Covtype_B, SUSY and Covtype_M datasets 100 times with noise from $N(0, 2)$. We treat the features duplicated from a feature of original sample as a group. For the RCV1 dataset, we partition the features into 48 groups. Note that, all datasets used in the experiments are with large scale both in sample size and feature dimensionality simultaneously.

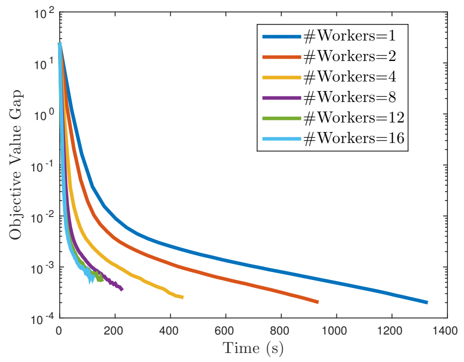
5.2 Experimental Results and Discussion

Figure 1 presents the convergence of objective values of DSPG, AsySPVR, AsySPVR $mb = 100$, AsyDSPG+ and AsyDSPG+ $mb = 100$ on the datasets of Covtype_B, RCV1, SUSY, Covtype_M and MNIST. The results confirm that our AsyDSPG+ is much faster than DSPG and AsySPVR. Because all the datasets used in the experiments are with large scale both in sample size and feature dimensionality simultaneously, Figures 1b and 1e show that our AsyDSPG+ scales

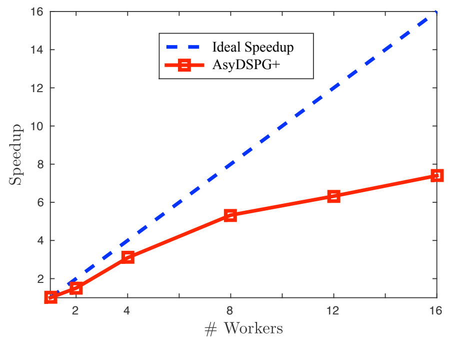
[1]The datasets are from `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.

Table 3: Summary of large-scale real-world datasets in our experiments.

Task	Dataset	Class	Features	Samples	Sparsity
Classification	Covtype_B	2	54	581,012	22 %
	RCV1	2	47,236	677,399	0.16 %
	SUSY	2	18	5,000,000	100 %
Regression	Covtype_M	7	54	581,012	22 %
	MNIST	10	784	1,000,000	100 %



(a) Objective value with different number of workers.



(b) Speedup results on MNIST dataset.

Figure 2: Speedup results of AsyDSPG+ on the MNIST dataset.

well with the simultaneously increasing of sample size and feature dimensionality.

To estimate the scalability of our AsyDSPG+, we perform AsyDSPG+ on 1, 2, 4, 8, 12 and 16 workers (*i.e.*, cores) to observe the speedup. Figure 2 presents the speedup results of AsyDSPG+ on MNIST dataset, which show that AsyDSPG+ can have a near-linear speedup on a parallel system with shared memory. Because we do not use any lock in the implementation of AsyDSPG+.

6 Conclusion

Existing (asynchronous) stochastic algorithms [Hong et al., 2013, Li et al., 2013, 2016, Meng et al., 2016] for solving the composite group regularized optimization problem (1) cannot scale well in sample size and dimensionality simultaneously. To address this challenging problem, in this paper, we propose a novel asynchronous doubly stochastic proximal gradient algorithm with the SVRG variance reduction technique (AsyDSPG+). To the best of our knowlege, AsyDSPG+ is the first asynchronous doubly stochastic proximal gradient algorithm, which can scale well with the large sample size and high feature dimensionality simultaneously. We prove that AsyDSPG+ achieves a linear convergence rate when the function f has the optimal strong convexity property, and a sublinear rate when f is with the general convexity or with the non-convexity. The experimental results on various large-scale real-world datasets not only confirm the fast convergence of our new method, but also show that AsyDSPG+ scales better than the existing algorithms with the sample size and dimension simultaneously. Meanwhile, a near-linear speedup of our AsyDSPG+ on a parallel system with shared memory can be observed.

In the future, we want to extend our asynchronous doubly stochastic proximal gradient algorithm and the theoretical analysis to the overlapping group regularized learning problems [Yuan et al., 2011], inexact proximal gradients descent algorithms [Bin Gu, 2018a] and composition problems [Zhouyuan Huo, 2018].

Acknowledgements

The work was partially supported by the following grants: NSF-IIS 1302675, NSF-IIS 1344152, NSF-DBI 1356628, NSF-IIS 1619308, NSF-IIS 1633753, NIH R01 AG049371.

References

Jinbo Bi, Senthil Periaswamy, Kazunori Okada, Toshiro Kubota, Glenn Fung, Marcos Salganicoff, and R Bharat Rao. 2006. Computer aided detection via asymmetric cascade of sparse hyperplane classifiers. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 837–844.

Yitan Li, Linli Xu, Xiaowei Zhong, and Qing Ling. Make workers work harder: Decoupled asynchronous proximal stochastic gradient descent. *arXiv preprint arXiv:1605.06619*, 2016.

Mathieu Blondel, Kazuhiro Seki, and Kuniaki Uehara. 2013. Block coordinate descent algorithms for large-scale sparse multiclass classification. *Machine learning* 93, 1 (2013), 31–52.

Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J Smola, Jing Jiang, and Chong Wang. 2014. Jointly modeling aspects, ratings and sentiments for movie recommendation (jmars). In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 193–202.

Mingyi Hong, Xiangfeng Wang, Meisam Razaviyayn, and Zhi-Quan Luo. 2013. Iteration complexity analysis of block coordinate descent methods. *Mathematical Programming* (2013), 1–30.

Cho-Jui Hsieh, Hsiang-Fu Yu, and Inderjit S Dhillon. 2015. Passcode: Parallel asynchronous stochastic dual co-ordinate descent. *arXiv preprint* (2015).

Yu-Tseh Chi, Mohsen Ali, Ajit Rajwade, and Jeffrey Ho. 2013. Block and group regularized sparse modeling for dictionary learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 377–382.

Thorsten Joachims and Adith Swaminathan. 2016. Counterfactual evaluation and learning for search, recommendation and ad placement. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 1199–1201.

Ji Liu and Stephen J Wright. 2015. Asynchronous stochastic coordinate descent: Parallelism and convergence properties. *SIAM Journal on Optimization* 25, 1 (2015), 351–376.

Ji Liu, Stephen J Wright, Christopher Ré, Victor Bittorf, and Srikrishna Sridhar. 2015. An asynchronous parallel stochastic coordinate descent algorithm. *Journal of Machine Learning Research* 16, 285-322 (2015), 1–5.

Zhaosong Lu and Lin Xiao. 2015. On the complexity analysis of randomized block-coordinate descent methods. *Mathematical Programming* 152, 1-2 (2015), 615–642.

Tuo Zhao, Mo Yu, Yiming Wang, Raman Arora, and Han Liu. Accelerated mini-batch randomized block coordinate descent method. In *Advances in neural information processing systems*, pages 3329–3337, 2014.

Horia Mania, Xinghao Pan, Dimitris Papailiopoulos, Benjamin Recht, Kannan Ramchandran, and Michael I Jordan. 2015. Perturbed iterate analysis for asynchronous stochastic optimization. *arXiv preprint arXiv:1507.06970* (2015).

Qi Meng, Wei Chen, Jingcheng Yu, Taifeng Wang, Zhi-Ming Ma, and Tie-Yan Liu. Asynchronous stochastic proximal optimization algorithms with variance reduction. *arXiv preprint arXiv:1609.08435*, 2016.

Atsushi Nitanda. 2014. Stochastic proximal gradient descent with acceleration techniques. In *Advances in Neural Information Processing Systems*. 1574–1582.

Meisam Razaviyayn, Mingyi Hong, Zhi-Quan Luo, and Jong-Shi Pang. 2014. Parallel successive convex approximation for nonsmooth nonconvex optimization. In *Advances in Neural Information Processing Systems*. 1440–1448.

Mu Li, David G Andersen, and Alexander Smola. Distributed delayed proximal gradient methods. In *NIPS Workshop on Optimization for Machine Learning*, 2013.

Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. 2011. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems*. 693–701.

Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex J Smola. 2015. On variance reduction in stochastic gradient descent and its asynchronous variants. In *Advances in Neural Information Processing Systems*. 2647–2655.

Peter Richtárik and Martin Takáč. 2016. Parallel coordinate descent methods for big data optimization. *Mathematical Programming* 156, 1-2 (2016), 433–484.

Volker Roth and Bernd Fischer. 2008. The group-lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In *Proceedings of the 25th international conference on Machine learning*. ACM, 848–855.

Mark Schmidt, Nicolas Le Roux, and Francis Bach. 2013. Minimizing finite sums with the stochastic average gradient. *arXiv preprint arXiv:1309.2388* (2013).

Shai Shalev-Shwartz. 2016. SDCA without Duality, Regularization, and Individual Convexity. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. 747–754. `http://jmlr.org/proceedings/papers/v48/shalev-shwartza16.html`

Shai Shalev-Shwartz and Tong Zhang. 2014. Accelerated Proximal Stochastic Dual Coordinate Ascent for Regularized Loss Minimization.. In *ICML*. 64–72.

Shirish Krishnaj Shevade and S Sathiya Keerthi. 2003. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics* 19, 17 (2003), 2246–2253.

Martin Takác. 2014. Randomized coordinate descent methods for big data optimization. (2014).

Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), 267–288.

Lin Xiao and Tong Zhang. 2014. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization* 24, 4 (2014), 2057–2075.

Xiyu Yu and Dacheng Tao. 2016. Variance-Reduced Proximal Stochastic Gradient Descent for Non-convex Composite optimization. *arXiv preprint arXiv:1606.00602* (2016).

Tong Zhang. 2004. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the twenty-first international conference on Machine learning*. ACM, 116.

Shen-Yi Zhao and Wu-Jun Li. 2016. Fast Asynchronous Parallel Stochastic Gradient Descent: A Lock-Free Approach with Convergence Guarantee. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Tuo Zhao, Mo Yu, Yiming Wang, Raman Arora, and Han Liu. 2014. Accelerated mini-batch randomized block coordinate descent method. In *Advances in neural information processing systems*. 3329–3337.

Hui Zou and Trevor Hastie. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67, 2 (2005), 301–320.

Lei Yuan, Jun Liu, and Jieping Ye. Efficient methods for overlapping group lasso. In *Advances in Neural Information Processing Systems*, pages 352–360, 2011.

Zhouyuan Huo and Heng Huang. 2017. Asynchronous Mini-Batch Gradient Descent with Variance Reduction for Non-Convex Optimization. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.* 2043–2049.

Bin Gu, Xin Miao, Zhouyuan Huo Heng Huang. 2018b. Asynchronous Doubly Stochastic Sparse Kernel Learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 2-7, 2018, New Orleans, Louisiana, USA.*

Zhouyuan Huo, Bin Gu, Ji Liu, Heng Huang. 2018. Accelerated Method for Stochastic Composition Optimization with Nonsmooth Regularization. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 2-7, 2018, New Orleans, Louisiana, USA.*

Bin Gu, De Wang, Zhouyuan, Huo Heng Huang. 2018a. Inexact Proximal Gradient Methods for Non-convex and Non-smooth Optimization. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 2-7, 2018, New Orleans, Louisiana, USA.*